Predictive learning by a burst-dependent learning rule

G. William Chapman[1*], Michael E. Hasselmo[1]

1 Center for Systems Neuroscience, Boston University, Boston, MA, USA

* wchapman@bu.edu

Abstract

Humans and other animals are able to quickly generalize latent dynamics of spatiotemporal sequences, often from a minimal number of previous experiences. Additionally, internal representations of external stimuli must remain stable, even in the presence of sensory noise, in order to be useful for informing behavior. In contrast, typical machine learning approaches require many thousands of samples, and generalize poorly to unexperienced examples, or fail completely to predict at long timescales. Here, we propose a novel neural network module which incorporates hierarchy and recurrent feedback terms, constituting a simplified model of neocortical microcircuits. This microcircuit predicts spatiotemporal trajectories at the input layer using a temporal error minimization algorithm. We show that this module is able to predict with higher accuracy into the future compared to traditional models. Investigating this model we find that successive predictive models learn representations which are increasingly removed from the raw sensory space, namely as successive temporal derivatives of the positional information. Next, we introduce a spiking neural network model which implements the rate-model through the use of a recently proposed biological learning rule utilizing dual-compartment neurons. We show that this network performs well on the same tasks as the mean-field models, by developing intrinsic dynamics that follow the dynamics of the external stimulus, while coordinating transmission of higher-order dynamics. Taken as a whole, these findings suggest that hierarchical temporal abstraction of sequences, rather than feed-forward reconstruction, may be responsible for the ability of neural systems to quickly adapt to novel situations.

1 Introduction

Neocortical circuits mediate a broad variety of cognitive functions, including the extraction of rules in different behavioral tasks (Bhandari and Badre, 2018; Zhu et al., 2018; Hasselmo and Stern, 2018; Wallis et al., 2001; Buschman et al., 2012). One aspect of the extraction of rules involves the tracking of dynamics of sensory stimuli(Yoo et al., 2020) and self-location as an agent navigates in an environment (McNaughton et al., 2006; Byrne et al., 2007; Hasselmo, 2005; Bicanski and Burgess, 2018). A number of different cortical regions are implicated in these types of functions, including parietal cortex(Byrne et al., 2007; Bicanski and Burgess, 2018), retrosplenial cortex(Alexander et al., 2020), entorhinal cortex(Brandon et al., 2013), and prefrontal cortex. Simultaneously, anatomical evidence suggests that there may be common features of cortical circuitry throughout different cortical regions(Douglas et al., 1989; Bastos et al., 2012; Mountcastle, 1997; Rockland, 2010). Given the distributed nature of tracking, as well as this anatomical consistency across cortical circuits, we hypothesize that particular aspects of cortical organization may be responsible for building accurate internal representations of these external stimuli. Here, we work towards building a model of cortical microcircuits which replicates this ability to predicatively code for trajectories of stimuli.

Previous Work There have been many models of prediction of time series, both from a machine learning perspective and a neurally inspired framework. While our goal is to create a biologically realistic model of prediction, we discuss machine learning (ML) approaches as well. These ML approaches serve as a baseline to which we can compare our model's performance, but also as extremely abstracted and mathematically optimized models of neural systems (Rumelhart and McLelland, 1986). The most common form of ML sequence prediction is sequence-to-sequence modeling, which utilizes backpropagation of errors through time (BPTT) (Williams and Zipser, 1989). While this approach has high success in areas such as natural language processing (Sutskever et al., 2014), they are essentially recurrent autoencoders, and typically fail when external teaching signals are inconsistent or sparse (Bengio et al., 1994). Alternative approaches based on echo-state (known as FORCE training) (Sussillo and Abbott, 2009), or liquid-state (Boerlin et al., 2013), networks are able to mimic external dynamics autonomously after a brief training period. FORCE and related methods however rely on a specific connectivity in which a decoded state is optimized by an external teacher and fed back into the network (Nicola and Clopath, 2017; Denève et al., 2017), and can not learn when the external teaching signal appears stochastically (see supplemental materials). In contrast, animals must perform in an environment where external stimuli may appear and disappear at random intervals, and building incorrect internal models of position is not sufficient for driving behavior. Prompted by this discrepancy, we consider three core aspects in which neural systems are thought to be organized, and consider how they may be essential for sequence prediction.

Hierarchy Proposals of cortical function have used hierarchical representations of information across different regions. These include many examples of function. For example, in progressing from caudal to rostral regions in the visual system different cortical regions mediate a hierarchical transition of coding level (Gilbert and Li, 2013; DiCarlo et al., 2012) of individual points of an image with the extraction of edges (V1), to the coding of higher order features such as movement (MT), color (V4) and ultimately the identity of an object such as a face (IT) (Desimone and Schein, 1987; Hasselmo et al., 1989; DiCarlo et al., 2012). Another example concerns position information in which the elements of self-movement can be extracted separately in terms of position (O'Keefe and Dostrovsky, 1971; O'Keefe and Burgess, 2005), or separately as velocity (Kropff et al., 2015; Hinman et al., 2016) or as acceleration (Kropff et al., 2021). This also applies to higher level cognitive control modeled on reinforcement learning in which sub-policies in posterior frontal cortex are controlled by higher level hierarchical control policies in more anterior regions (Koechlin and Summerfield, 2007; Badre and Frank, 2012; Badre and D'Esposito, 2009). In modeling, these types of hierarchical representations are used in hierarchical reinforcement learning (Sutton et al., 1999) and in hierarchical Bayesian coding (Kingma and Welling, 2019). Here we build on the idea of hierarchical representations of cortical regions to form a predictive representation of dynamics, where the higher cortical regions track successively derived features.

Supervised vs Predictive Coding Previous approaches tend to rely on supervised learning, in which a whole or portion of a sequence is provided at one 'end' of a network, and a prediction is formed at the other end of the network, which is then used to update synaptic weights. In contrast, a number of cortical models involve generative-predictive coding (Rao and Ballard, 1999). In this framework, a given cortical region receives a feedforward (eg: sensory) signal, and a feedback signal consisting of the expected feedforward signal. Each region then calculates the difference between the feedforward (true) and feedback (expected) activity, which is then passed to the next cortical region (Bastos et al., 2012). In the example of early visual cortex, these feedback signals may be a line segment, while feedforward signals then carry the spatial error, giving rise to the end-stopping phenomenon and other experimental findings (Lotter et al., 2020). While these forms of predictive coding hypothesize that the feedforward activity represents prediction errors, other studies have hypothesized that there is no such explicit computation of error. These alternative forms of predictive coding pose a feedforward signal which is, in each cortical region, simply a transformed version of it's own input, and feedback signals represent some latent feature which may be informative for improving these lower-level predictions (O'Reilly et al., 2021). A common aspect to both the feedforward error and feedforward prediction frameworks is that reconstruction or prediction of the external stimulus occurs at the lowest cortical regions. This is similar to Helmholtz Machines, in which external stimuli drive the formation of self organizing maps, and feedback weights create generative biases that can reconstruct stimuli from scratch or distorted signals (Dayan et al., 1995). More recent work has also begun to investigate how the lower-level reconstruction approach can improve or simplify temporal prediction in machine-learning contexts (Gregor et al., 2014; Sutskever et al., 2009). In addition to *hierarchical* prediction, explicit prediction over time has also been shown to create compressed representations of stimuli, whereas non-predictive autoencoders do not (Recanatesi et al., 2021). Inspired by the success of hierarchical predictive coding, in both biological plausibility and successful stimulus prediction, we look for a model which incorporates a predictive and generative framework. Specifically, we expect that some temporal features of stimuli, such as temporal derivatives, may be derived in a hierarchy, thus forming the basis for a temporally changing context which can improve lower-level predictions.

Learning Rules Most of the articles cited up until this point rely either on some form of backpropagation through time (BPTT), or an abstracted form of statistical optimization (eg: (Bastos et al., 2012)). However backpropagation is biologically implausible, in its generally presented form. There have been several proposed learning rules which approximate backpropagation in a more biologically plausible form, such as Contrastive Hebbian Learning (O'Reilly, 1997), Feedback Alignment (Lillicrap et al., 2016), or burst propagation networks (Payeur et al., 2021). However, these studies tend to focus on supervised learning problems, in which a series of feedforward regions attempt to match a given input to a desired output label. Even if the architecture of the network is modified such that reconstruction is performed at the lowest level, BPTT is not guaranteed to converge on optimal weights (as we show in our section 'Intermediate Models' below). In contrast, a more biologically plausible and local learning rule may converge on more optimal weights, despite being worse for generating universal function approximations, if it is more suited towards directly minimizing temporal error. We therefore propose to investigate a learning rule which incorporates both feedforward and feedback weights to minimize errors over time, and then connect it to a spike-based learning rule from experimental and computational literature (Payeur et al., 2021).

Scope of Work In the current work we focus on training networks to predict deterministic dynamical systems for long periods after stimulus has stopped being presented. We begin by illustrating how training by backpropagation through time fails in the presence of unreliable external input. We then present three sequential modifications to the baseline architecture, each of which is based on the biological principles discussed above. We then present a final modification in the introduction of a local learning rule, inspired by theories of error-driven learning in neocortex. We show that this biologically inspired "predictive module" performs significantly better at predicting dynamics over long timescales. Investigating the learned representations in the predictive module, we find that successive modules learn to encode successive temporal derivatives. Finally, in the third portion of the results section, we present a spiking adaptation of the predictive module. This spiking model utilizes a set of biologically plausible learning rules, and performs with similar accuracy and feature learning as the continuous approximation.

2 Methods

2.1 Task

In order to test the degree to which our models can predict underlying dynamics of an external stimulus, we utilize the commonly used sum of sinusoids task (Fig 1A) which evolves according to equation 1 (Duggins and Eliasmith, 2022; Sussillo and Abbott, 2009), with two modifications. First, in comparison to earlier papers, we generate the task in a procedural manner, where the relative phase and frequency of the underlying task change on each presentation (see table 5 for parameter ranges). By altering these parameters on each trial, we test the ability of the networks to learn general dynamics, rather than forecasting from previously seen histories. Secondly, we utilize a paradigm known as teacher forcing in which, over the course of training, the interval at which the network observed the ground-truth stimulus decreases. This frequency of observation of ground-truth, termed 'teaching ratio', is decreased over training according to a hyperbolic schedule (Fig 1E). On frames where the network is able to observe the ground-truth, the stimulus is presented to the lowest level of the network (Fig 1B, top). On frames where the network is not able to view the stimulus ("rollout"), the lowest region is instead presented with a decoded input from the previous time-step's activity (Fig 1B, Middle), or no input at all (Fig 1B Bottom). During the initial phases of learning, having a high teaching ratio ensures that the ground truth and input follow similar statistical distributions, and is necessary to ensure learning during early phases of training. As training progresses, predictions a few time steps out are likely to follow the same distribution as the ground truth, as the network predicts each next-frame with high accuracy. On any given trial, the network is guaranteed to receive the true stimulus input for the first half (150) of frames, in order to allow appropriate historical observations to propagate through the network. On each epoch, we test the network on an unseen set of trials, and measure performance using the teaching ratio according to the schedule described above (henceforth the 'local ratio'), and also using a teaching ratio of zero (Fig 1C). This allows investigation of how the networks learn to respond to the task which they are being trained to (local teaching ratio), while simultaneously investigating how they respond to long periods without external feedback.

$$P(t) = a_1 \sin(f_1 t + \phi_1) + a_2 \sin(f_2 t + \phi_2) \tag{1}$$

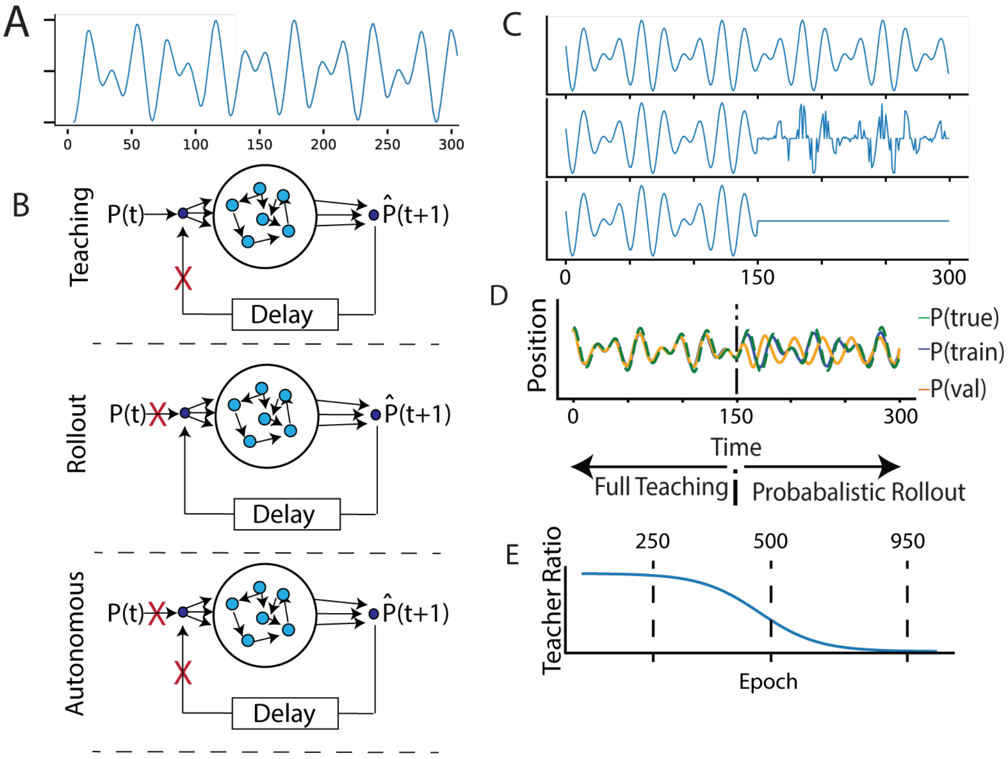

Fig 1. Overall setup for training of models. **A** Example of the task used, showing the simulated trajectory as a function of time. Trajectories as parametrically generated according to equation 1. **B** Different approaches to time-series prediction. (Top) During a teaching frame, a network receives the ground truth position, and estimates the next frame. No information about the decoded prediction is returned to the network. (Middle) On a rollout frame, the network does not receive the ground truth, and instead receives its previous decoded prediction. (Bottom) In alternative frameworks, used later in the paper, a network may evolve autonomously, receiving neither the ground truth nor the decoded prediction. **C** Example of a target signal (top), and the corresponding *external* signal to the networks. (Middle) Throughout training, the network is presented with a training signal which is equal to the target signal for the first half of the trial, but randomly set to zero during the probabilistic rollout period (p=0.50 in example). On frames where the external signal is zero, the network evolves according to the corresponding rules from B. (Bottom) Throughout training an additional 'validation' signal is presented, in which the rollout teacher ratio is set to zero, but no learning occurs; this shows how the network would perform at the fully autonomous task at any point in training. **D** Example of a trial for a backpropagation through time (BPTT) network partly through training. During the full teaching period the network closely matches the target signal (green). During probabilistic rollout (blue), the network continues to closely approximate the target signal; the validation configuration (orange) diverges from the target. Note that while we use a signal frozen example in this and future traces, the actual signal given to the network is randomly generated on each trial. **E** The teacher ratio decreases over the course of training. Dashed lines indicate points where examples are drawn from in subsequent figures.

2.2 Simulation Tools and Model Training

In the following sections we introduce a number of models which progress from a simple baseline model to a final novel architecture and learning rule. In each section equations which model the dynamics and weight updates are introduced, and parameters and initial distributions are set in tables. All models, with the exception of the spiking model, are implemented in PyTorch (Paszke et al., 2019), and evolve according to a simple forward Euler method, utilizing a timestep of 1. In all cases, error is calculated as the sum of mean-square errors across the entirety of a trial. All models before the predictive module are optimized by backpropagation through time (BPTT), using the ADAM optimizer (Kingma and Dhariwal, 2018). In the predictive module section we replace BPTT with a novel learning rule which utilizes only local variables, and update weights in the same manner and frequency that activities are updated. Hyperparameter optimizations are found in table 4 and were chosen by an exhaustive grid search, representing 36-48 configurations for each model. For each hyperparameter configuration, five separate randomly models were run and their minimum local error or validation error were averaged together. The minimum error for each models best hyperparameter configuration are given in table 4 The spiking model was implemented in BindsNET (Hazan et al., 2018), a spiking neural network simulator built in Python, and utilize a timestep of 0.1ms, and is described in more depth in the corresponding section below.

2.3 Baseline Model

As a comparison for our biologically-inspired network described later, we implement a standard sequence-to-sequence model. This baseline model consists of a linear encoder, a number of hidden Elman RNN layers, and a linear decoding layer. The activity of each layer, denoted 'R(t)' evolved according to the equation:

$$\begin{aligned} R_0(t) &= \tanh(W_{I0}I(t) + b_0) \\ R_i(t) &= \tanh(W_{ii}R_i(t-1) + W_{(i-1)i}R_{i-1}(t) + b_i) \\ R_N(t) &= W_{(i-1)i}R_{N-1}(t) + b_N \end{aligned} \tag{2}$$

Where subscripts denote either layer or weight (PrePost). On each time-step, the external input I is either the ground-truth stimulus (Fig 1B, top), or previous decoded output (Fig 1B, middle), according to the teaching ratio (Fig 1C and Fig 1E). We also investigated the performance of a similar architecture but instead utilizing LSTM units.

2.4 Intermediate Models

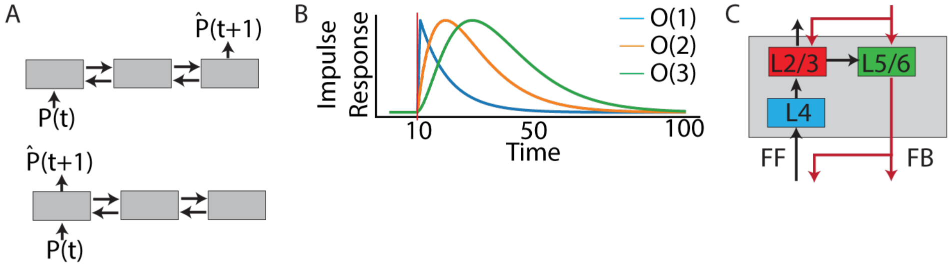

Fig 2. Intermediate models, showing stepwise changes towards biologically inspired architecture. **A** Stacked RNN: In the baseline model activity entered a stack of Simple RNNs at the base and propagates to a top layer where readout occurred. In the stacked RNN case, there are bidirectional connections between each hidden layer, and readout occurs either at the top layer (top) or input layer (bottom). **B** Leaky integrators (LI): Instantaneous responses in the simple RNNs are now replaced with leaky integrators which have a finite-time response to inputs. Here we show the impulse response of a single (blue) LI, two LIs in series (orange) and three LIs in series (green). Each successive layer introduces an additional delay (offset) and smoothing of the input. **C** We now introduce separate feedforward (FF) and feedback (FB) pathways. Each colored block represents a set of leaky integrators, now arranged in a manner similar to neocortical motifs (see text), allowing separate feedforward (black) and feedback (red) pathways. Feedback connections project to L2/3 and L5/6 of lower regions.

We next introduce a series of intermediate models, each of which incorporates one of the principles outline in the introduction. The purpose of these models is to introduce aspects of the final proposed model, while allowing investigation into how each of these architectural or dynamical principles influence the ability of a standard learning rule (BPTT) to learn in the presence of our task. For each of these changes, we provide a biological rationale, and a normative explanation for how the change may improve performance.

Recurrence & Readout Layer Compared to the feed-forward networks of the baseline approach, cortical regions tend to be bidirectionally connected. For this reason our first intermediate model utilizes a series of stacked Elman RNN layers but adds an additional weight from each layer back to its preceding layer (Figure 2 A, top). Next, we address the issue of where signals are reconstructed. In a hierarchical circuit the further we move up a hierarchy, the less information about the external stimulus is directly available. A common approach in machine learning settings is to introduce a ‘U’ like structure or provide skip connections, which bypass intermediate transformations and provide direct routes for lower-level information to be integrated into higher-level (deeper) regions. Here, we take a slightly different approach, inspired by biological models of predictive coding. Instead of providing skip connections that bring low-level information to higher regions, we provide backwards-projections, such that the representation from a deeper region influences the activity of earlier layers. This achieves the same result that less processed and more processed information integrate in a given layer, but creates several patterns of activity that have been observed in neural data (Lotter et al., 2020) and satisfies the general framework of predictive coding in which deeper layers represent some information not present in earlier layers and project that backwards (Rao and Ballard, 1999). The result is a stacked hierarchy of RNN cells, but reconstruction now occurs at the lowest level of the model (Fig 2 A, bottom). Mathematically, both of these scenarios can be expressed as:

$$\begin{aligned} R_0(t) &= \tanh(W_{00}R_0(t-1) + W_{10}R_1(t-1) + W_{I0}I(t) + b_0) \\ R_i(t) &= \tanh(W_{ii}R_i(t-1) + W_{(i-1)i}R_{i-1}(t-1) + W_{(i+1)i}R_{i+1}(t-1) + b_i) \\ R_N(t) &= \tanh(W_{NN}R_N(t-1) + W_{(N-1)N}R_{N-1}(t-1) + b_N) \\ R_{out}(t) &= W_{X-out}R_X(t) + b_{out} \end{aligned} \tag{3}$$

Where “X” denotes either the bottom-most or top-post layer, depending on the condition, and R_{out} consists only of a single unit. In order for each layer to update synchronously, inputs to all layers are collected before any layer's activity is updated. This results in a 1-frame delay between each layer (2 frames from bottom layer to top layer, 4 frames for bottom layer activity to propagate all the way to the deepest layer and then back to the lowest layer).

Leaky Integrator Units From a biological perspective, the activity of a population of neurons can not change instantaneously, and is often modeled as a leaky-integrator (LI), in which the potential of a single unit decays to zero in the absence of any input, according to the equation:

$$\begin{aligned} \tau\frac{dv_L}{dt} &= -v_L(t) + \sum_{N\in A} W_{NL}R_N(t) \\ R_L(t) &= \tanh(v_L(t)) \end{aligned} \tag{4}$$

Where τ represents a slow-leak time constant (set to 10 frames), v represents the 'membrane potential', r represents the outgoing activity, and A is the set of all other layers which project to layer L, again following the same bidirectional connectivity pattern from above. Leaky integrators have also been shown to be useful in generation of complex trajectories by providing a temporal reservoir of past activity (see supplemental material). Functionally, LI units act as a low-pass filter, which can filter out transiently absent inputs to a certain degree, and may be useful in smoothing out small errors in rollout dynamics. An additional consequence of the low-pass function is that input to a given unit has a *delay* before causing maximal changes in a downstream unit (Fig 2 B). This delay means that in our stacked RNN setting, changes that occur in a given layer and propagate into hidden layers and back have a $2\tau\Delta N$ delay before they can be fully propagated and integrated into the layer. This means that in order for a signal to be useful by the time it propagates back, there must be some explicit bias towards prediction at each synaptic step.

Separate Feedforward and Feedback Pathways For the final intermediate model, we increase the connective complexity within each region. In neocortex there is a distinct laminar organization in which there is a separate feedforward pathway (lower regions →granular →superficial →higher regions) and feedback pathway (higher regions →infragranular →lower regions) (Haeusler and Maass, 2007). Compared to a simple stacked approach, this branching connectivity allows infragranular neurons to integrate feedforward and feedback signals before passing them back down the cortical hierarchy, and has been suggested to be important for predictive coding approaches (Lotter et al., 2017). When combined with leaky-integrator units, this integration of paths also occurs over time, since feedback pathways represent older representations than feedforward ones. In order to test whether this separation of pathways can be utilized for prediction of external stimuli we modify the previous intermediate stacked model such that each region contains these separate sub-modules. The overall connectivity pattern is in Table 1, or illustrated in figure 2 C, and each sub-module continues to follow the dynamics of equation 4.

	W$_{PrePost}$
Recur	W_{GG} W_{SS} W_{II}
Intra	W_{GS} W_{SI}
Inter	W_{SG} W_{IS} W_{II}

Table 1. Organization of connections in the laminar intermediate model. Weights are indicated as PrePos. G is "granular" (lamina 4), S is "superficial" (Lamina 2/3) and I is "infragranular" (Lamina 5/6). Interegional SG is the primary feedforward pathway (projects to higher regions), while IS and II project from higher regions to lower ones, in accordance with general neuroanatomical studies.

2.5 Predictive Module

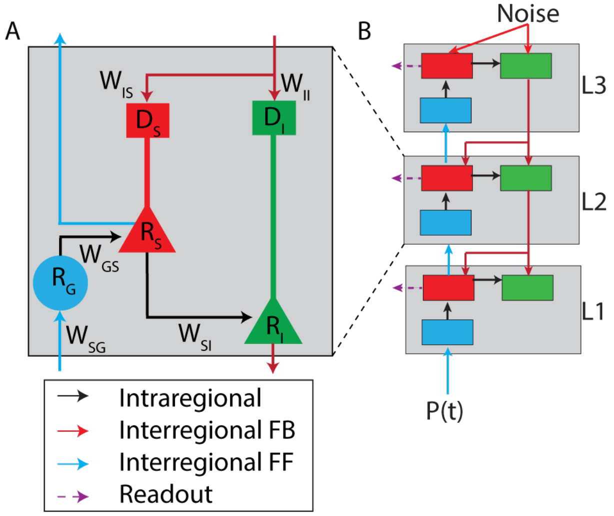

Fig 3. Architecture of the final predictive module. **A** The structure of each module, where rates ('R') of each submodule represent the firing rate of a given population of neurons, and weights are labeled according to their source and target submodules. Units are coloured by their laminar location, also denoted by subscript (G)ranular, (S)uperficial, (I)nfragranular, and (Distal). Weights are coloured by according to their primary function in the feedforward (cyan), feedback (maroon) or local (black) pathways. **B** Zooming out to show connectivity among multiple predictive modules. For the lowest module feedforward activity comes from the external stimulus P(t), while for higher regions it comes from lower level superficial neural activity (R_S of A). For the highest level, feedback activity comes from an OU process representing background fluctuations, while lower regions receive it from higher level infragranular activity (R_I of A).

The goal here is to introduce a learning rule which can effectively utilize the separate feedback pathway to self-supervise learning in lower cortical regions. In order to reach this we perform three key modifications

Dual-Compartment Units Results from the intermediate models (see results) indicate that in the BPTT approach having a branching feedback pathway decreases the accuracy of backpropogation based credit assignment. However, several lines of research have suggested that the feedback component of cortical microcircuits is critical for guiding learning in biological systems (Magee and Grienberger, 2020; Larkum, 2013; Greedy et al., 2022). A common thread in each of these approaches is that the feedback pathway terminates on the distal dendrites of pyramidal neurons, and provides a mechanism through which to gate traditional Hebbian-like learning on the feedforward path synapses. Thus, in order to implement a similar rule, we modify the superficial and deep units of our model to implement a mean-rate version of the dual-compartment spiking pyramidal model. Each pyramidal neuron now contains an additional term in their dynamics:

$$\begin{aligned}
D_L(t) &= \tanh(\sum W_{ID_L} R_I(t)) \\
\tau \frac{dv_L}{dt} &= -v_L(t) + D_L(t) + \sum_{N \in A} W_{NL} R_N(t) \\
R_L(t) &= \tanh(v_L(t))
\end{aligned} \tag{5}$$

Here, the first term indicates the distal dendritic potential for a given unit, which is follows its own activation function based on the sum of feedback activities R_I The second term specifies that the 'soma' of the dual compartment model follows the same leaky-integrator dynamics of equation 4, but also receives a 1:1 input from its own distal component, representing passive intracellular coupling of these compartments.

Learning Rule Previous models of biological learning have suggested that the distal dendritic potentials (D_L) can guide learning by changing either burst rate (Payeur et al., 2021) or intracellular calcium levels (Bittner et al., 2017; Larkum, 2013). We will model the burst-based rule explicitly in the spiking model, but for the rate model implement this simply as a unit-by-unit gating of learning rate. Each unit then attempts to modify the temporal error in membrane potential ($E_{post}(t)$) by modifying the weight in proportion to the amount of presynaptic and postsynaptic activity. We then identify the overall goal of the network, which is to minimize temporal discrepancies in the sum of inputs. Given the temporal error, and a feedback signal for guiding learning, we propose a three-factor learning rule:

$$\begin{aligned}
E_{post}(t) &= (v_L(t) - v_L(t-1)) \\
\Delta W_{PrePost} &= \eta r_{pre}(t) \otimes D_L(t) \odot E_{post}(t)
\end{aligned} \tag{6}$$

Where $\odot$ represents the Hadamard product, $\otimes$ the outer product, and η is a learning-rate (set at 0.01). Here $E_L(t)$ represents the overall objective of the network, which is to minimize the first order temporal error in unit potential, which appears as $E_{post}(t)$ in the weight update. The outer product of the presynaptic and postsynaptic terms is identical to the associative term found in many learning rules (Kohonen, 1982), but now contains an additional 'pushing' term in the minimization of temporal change (Greedy et al., 2022), and further modulated by the degree of feedback prediction ($D_L(t)$). Weights were initialized as shown in table 2, and learning occurred on every time step, regardless of whether the frame was forced or rollout, with the exception of validation trials.

Dynamic Vs Static Connections The learning rule proposed above is reliant on a feedback gating signal as well and has an explicit term to modify weights in order to minimize feedforward prediction errors (Brea et al., 2016), posing two issues. First, because the learning rule is locally greedy, there is a global minimum error of all synapses reach zero, resulting in a constant zero error-term. Secondly, the gating term is reliant on the distal compartment of pyramidal units, indicating that the rule can not apply to the granular units. However, experimental evidence has suggested that error-driven gating primarily occurs in the pyramidal neurons and other physiologically similar populations. We therefore implement the granular layer as a reservoir, and do not update the weights terminating onto these layers (see Table 2). When combined with random initial conditions (see below) the reservoir dynamics allow the granular layers to avoid the global minimum of zero activity, and push other layers from the same region. Additionally, we leave the feedback weights, terminating on the distal compartments at their initial conditions, as the calcium or burst-dependent inspiration for the gating term is not valid for these synapses. Pragmatically, this means the predictive module implements a form of feedback alignment (Lillicrap et al., 2016). Overall, it is only the feedforward synapses onto pyramidal units that are expected to undergo the *error-driven* learning we investigate here.

	Connection$_{pre-post}$	Updates?	Initial Distribution
Recur	W_{XG-XG}	No	$Gauss(0, \frac{1}{2\sqrt{N}})$
	W_{XS-XS}	Yes	$Gauss(0, \frac{1}{2\sqrt{N}})$
	W_{XI-XI}	Yes	$Gauss(0, \frac{1}{2\sqrt{N}})$
Intra	W_{XG-XS}	Yes	$Unif(-\frac{1}{2\sqrt{N}}, \frac{1}{2\sqrt{N}})$
	W_{XS-XI}	Yes	$Unif(-\frac{1}{2\sqrt{N}}, \frac{1}{2\sqrt{N}})$
Inter	W_{XS-YG}	No	$Unif(-\frac{1}{2\sqrt{N}}, \frac{1}{2\sqrt{N}})$
	W_{YI-XS}	No	$Gauss(0, \frac{1}{2\sqrt{N}})$
	W_{YI-XI}	No	$Gauss(0, \frac{1}{2\sqrt{N}})$

Table 2. Initialization of weights in the predictive module. Weights are denoted as PreregionPreLamina-PostregionPostLamina, where X indicates a given region, and Y is X+1; G is "granular" (lamina 4), S is "superficial" (Lamina 2/3) and I is "infragranular" (Lamina 5/6). Feedback weights (going from Y to X) indicate connections targeting the postsynaptic learning-gate, rather than the strong somatic potential.

Noise There are two sources of noise in the model we implement here. First, on each trial the activity of all units (and distal compartments) is randomly initialized. Secondly, for the upper-most module there is no higher level to provide feedback activity to the distal dendrites, and we replace this instead with a small Ornstein–Uhlenbeck (OU) process of mean zero, τ of 2 timesteps, and σ of 0.05.

Readout Because the predictive module does not directly optimize for readout of the target signal by a readout weight, we instead optimize readout weights (W) for each layer of the predictive module by constrained least-squares ('Ridge' in sklearn) such that the error term:

$$E = \sum (Y(t) - WR(t))^2 + \lambda ||W||_2^2 \tag{7}$$

was minimal, where $Y(t)$ is the signal we are attempting to decode, and λ is a constant (0.01) which penalizes large readout weights. Weights were optimized by a 5-fold cross-validation from the full forcing period, and reused to create the decoded signal.

2.6 Spiking Model

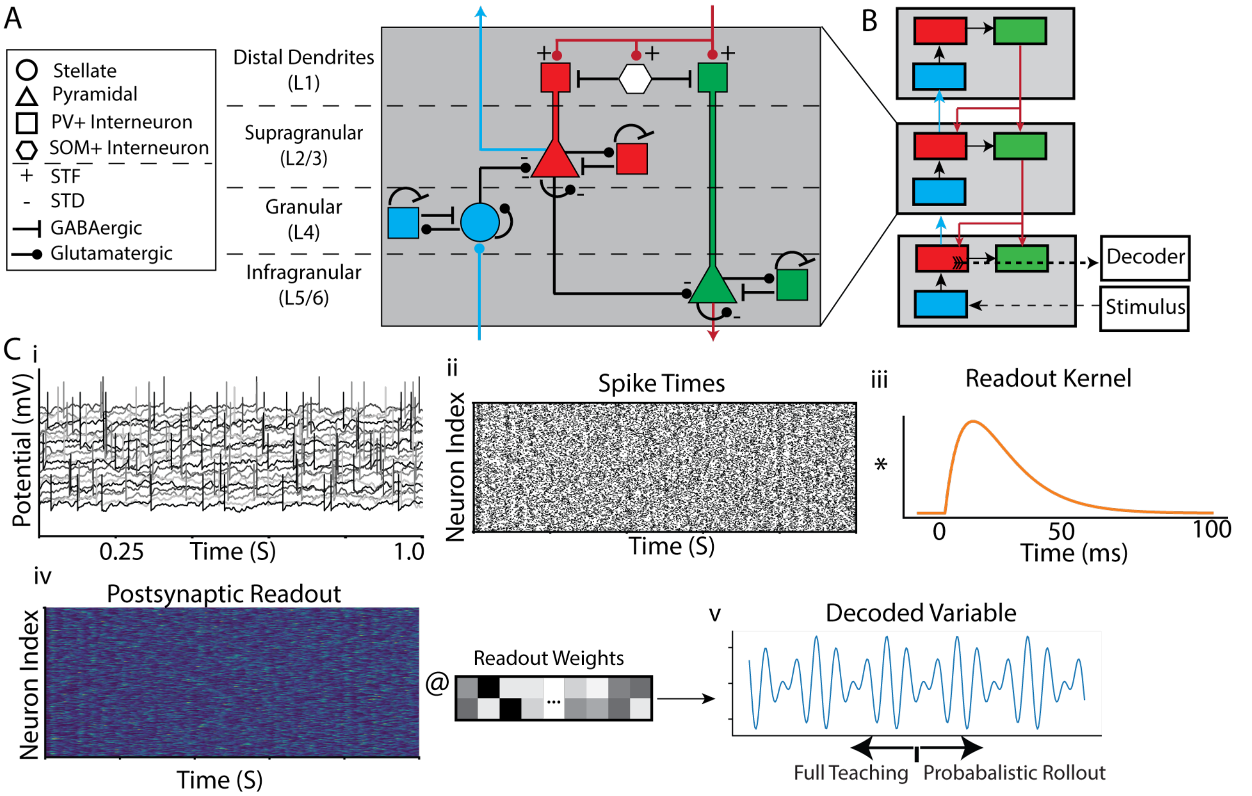

Fig 4. Spiking Model Architecture Each box represents a cortical region, and coloured blocks represent lamina (matching those in Fig 3). L2/3, L4, and L5/6 each contain an excitatory population (Pyramidal and stellate, as triangles/circles), along with a loosely balanced inhibitory population (parvalbumin interneurons, squares). Layer L1 consists only of an inhibitory population of somatostatin neurons, and the distal dendrites of layer L2/3 and L5/6 pyramidal neurons. Feed-forward activity begins in layer L4, propagates to layer L2/3, and then diverges to intra-columnar L5/6, and inter-columnar L4. Feedback activity travels from L5/6 to inter-columnar L1, where it guides burst rate of upstream pyramidal neurons. **B** The full spiking model, where each grey block is as in A. External stimuli are encoded according to a Poisson process and randomly projected into L4 of the lowest cortical region. The activity of each region is then read out with an independently trained decoder. **C** Structure of the decoder in. For a given lamina we record the membrane potential from all excitatory neurons (i), from which we extract spike trains (ii). The spike train is convolved by an idealized postsynaptic potential(iii) to give an output signal (iv) which is read out by optimized weights (W) to give the reconstructed signal (v).

Next, we implemented a spiking neural network, trained to perform the same sum-of-sinusoids task set as above. As with the predictive module, the repeating anatomical connective motifs of this model are inspired by the canonical cortical microcircuit (Douglas et al., 1989). Now however, we introduce several additional components such as inhibition and short-term potentiation, that are necessary for translation to a spiking setup. Each introduced set of parameters follows general findings from neuroanatomical studies. All spiking simulations are performed in BindsNET and utilize a simple forward method (timestep of 0.1ms) to implement the dynamics described in the equations below. On each timestep all of the incoming activities to each neural population are collected before any updates are made. Each population of neurons then updates according to its local dynamics before weights are updated according to short-term or long-term updates.

Overall structure As shown in 4A. Each region of the spiking model consisted of 3 lamina: granular, supragranular, and infragranular, similar to the predictive module. Now however, each of these three lamina contained a population of excitatory neurons, modeled as stellate cells for granular, and pyramidal neurons for supragranular and infragranular, along with a loosely coupled population of parvalbumin (PV) interneurons. All excitatory populations consisted of 4000 units, and inhibitory populations consisted of 1000 units, reflecting proportions found in-vivo. An incoming signal first passes through the granular layer, to supragranular, where it then diverges to the local infragranular neurons and the granular neurons of the next cortical region. Feedback activity travels from infragranular neurons to the distal dendrites of superficial and deep layers of the previous cortical region (Larkum, 2013). Connection patterns were initialized based on experimental findings (Haeusler and Maass, 2007), summarized in Table 3. Connection weights were uniformly distributed at +/- 10% of the mean weight given in the table, and 1-p of those weights, were set to zero.

Short Term Plasticity Short term plasticity (STP) was modeled in all synapses according to the Markram Tsodyks model (Markram et al., 1998), modeling depletion of neurotransmitters and presynaptic calcium potentiation:

$$\begin{aligned}\frac{dR}{dt} &= \frac{1-R(t)}{D} - u(t)R(t)\delta(t-t_{spike})\\ \frac{du}{dt} &= \frac{U-u}{F} - f[1-u(t)]\delta(t-t_{spike})\end{aligned} \tag{8}$$

STP parameters for each type of connection are summarized in table 8, and are chosen to match short-term facilitation (STF) or short-term depression (STD). The purpose of the STP dynamics are two-fold. First, they regulate the overall level of activity in the network, even as long-term weights are modified by the learning rule described below. Secondly, they allow a filtering of quick (STD) or slower (STP) spiking activities, as described in-depth in (Naud and Sprekeler, 2018). This second attribute is important for the learning rule described below.

Pre	Post	$\|\|W\|\|$ (pA)	p	Delay (ms)	STP	LTP	Remove STP
L4E	L4E	29	0.17	2	EE	None	N
L4E	L4I	60	0.19	1	EI	None	N
L4I	L4I	-41	0.5	1	II	None	N
L4I	L4E	-23	0.1	1	IE	None	N
L2/3E	L2/3E	45	0.26	2	EE	BDP	Y
L2/3E	L2/3I	50	0.21	1	EI	None	N
L2/3I	L2/3I	-36	0.25	1	II	None	N
L2/3I	L2/3E	-17	0.16	1	IE	None	N
L5/6E	L5/6E	45	0.09	2	EE	BDP	Y
L5/6E	L5/6I	24	0.1	1	EI	None	N
L5/6I	L5/6I	-32	0.6	1	II	None	N
L5/6I	L5/6E	-32	0.12	1	IE	None	N
L4E	L2/3E	60	0.28	2	EE	BDP	Y
L2/3E	L5/6E	37	0.55	2	EE	BDP	Y
L5/6E	L1D	15	.2	1	F	None	Y
L5/6E	L1I	15	.2	1	F	None	Y
L1I	L1D	-15	.2	1	D	None	Y

Table 3. Connectivity between lamina and regions in the spiking model. Weights were initialized to the mean weight given in this table, with probability p, then allowed to evolve. STP column refers to the parameters given in table 8. For pre and post populations, E refers to excitatory population somatic compartments, I refers to inhibitory somatic compartments, and D refers to distal dendritic components (pyramidal neurons only). The final column indicates which STP dynamics are removed for the biological necessity experiments.

Neural Dynamics As with other recent studies, pyramidal neurons were implemented with two compartments, representing the somatic and distal dendrites: 302 303

$$\begin{aligned} C_s\frac{dV_s}{dt} &= \frac{C_s}{\tau_s}(V_s - E_L) + g_s f(V_d) + I_s - w_s \\ \frac{dw_s}{dt} &= \frac{-w_s}{\tau_s} \end{aligned} \tag{9}$$

$$V \geq V_{Th} \rightarrow \begin{cases} V(t_+) &= V_{reset} \\ w(t_+) &= w(t_-) + b \end{cases} \tag{10}$$

Where $f(V_d)$ is represents the voltage/calcium gated channels in the dendritic compartment 304

$$f(V_d) = \frac{1}{1 + e^{-(V_d - E_d)/D_d}} \tag{11}$$

g_s is then the passive coupling parameter from the distal dendrites to the soma. 305

And for the dendritic compartment

$$\begin{aligned} C_d \frac{dV_d}{dt} &= -\frac{C_d}{\tau_d}(V_d - E_L) + g_d f(V_d) + c(S(t)) + I_d - w_d \\ \frac{dw_s}{dt} &= -\frac{w_d}{\tau_d} + a_w(V_d - E_L) \end{aligned} \tag{12}$$

Here, $S(t)$ represents the backpropagating action potential from the somatic compartment, which takes the form of a 2ms long pulse, delayed by 0.5ms from the time of the somatic spike($\delta(t)$):

$$S(t) = \int_{t-2.5ms}^{t-0.5} \delta(t)dt(\max 1) \tag{13}$$

This dual compartment approach allows separate control of firing rate and burst-rate, enabling multiplexing of feedforward and feedback activity. Similar to the D_L term from the predictive module, the distal dendritic compartment is receives feedback activity, and is responsible for controlling the learning rate of the somatic compartment, through rules described below (Payeur et al., 2021). In order to maintain the overall level of bursting in these neurons, an additional somatostatin (SOM) population of neurons was added which receives the same feedback activity as the distal dendrites, and provides a level of normalization keeping the burst rate approximately linear to the overall level of feedback activity (Vercruysse et al., 2021). The spiking properties of neurons in the network are illustrated in part B and C of Fig 4. All other neurons were implemented with adaptive exponential integrate and fire models (see section 6.3.1), with parameters to match their physiology (Naud et al., 2008).

2.6.1 Inputs and Readout

Each compartment received a multi-component input current, consisting of an external stimulus-dependent component, a synaptic component, and a stochastic background noisy input.

$$I_i = I_i^{ext} + I_i^{BG} + I_i^{syn} \tag{14}$$

The synaptic current I^{syn} is calculated as:

$$\frac{dI_{ij}^{syn}}{dt} = -I_{ij}^{syn}/\tau_{C_{ij}} + W_{ij} R \delta_j(t - d_{ij}) \tag{15}$$

Here, the spike train of presynaptic action potentials is delayed by a connection specific period d_{ij} before being modified by the short term plasticity term R and scaled by the synaptic weight matrix W. This modified impulse train is added at each point to a low-pass filter of the postsynaptic current, τ (5ms for inhibitory connections and 1ms for excitatory connections) allowing the postsynaptic synaptic potential (PSP) to follow the typical double-exponential pattern. The background input is necessary to keep units in the same lamina in a decorrelated state. This background was modeled separately for each compartment as an OU process, with parameters specific to each compartment type (see table 9). The background input was separately modeled as an OU process for each compartment:

$$\frac{d}{dt}I_i^{BG} = \frac{\mu - I_i^{BG}}{\tau^{OU}} + \sigma\sqrt{2/\tau^{OU}}\epsilon_i \tag{16}$$

Finally, the external input was zero for all units, with the exception of granular units in the lowest cortical region. For this region we encoded the external position as a Poisson random variable and processed this in the same manner as other synaptic inputs:

$$\begin{aligned} \delta(t) &\sim Pois(P(t)) \\ \frac{dI^{ext}}{dt} &= -I^{ext}/\tau_{C_{ij}} + W_{Input-4}\delta(t) \end{aligned} \tag{17}$$

Where $W_{Input-4}$ is initialized according to the distribution patterns for intraregional weights onto the granular lamina in table 3

2.6.2 Learning Rule

We next translate the predictive module learning rule to a spiking version, which reflects the recently proposed Burst-propagation rule (Payeur et al., 2021). As with the predictive module, we are only modeling the 'prediction-driven' aspects of learning, that is the portions of learning which are driven by top-down expectations which travel along the activity of infragranular activity to distal dendrites. The feedback pathway (L5/6-to-Distal dendrites) remained static, and this model therefore implements a version of feedback alignment (Lillicrap et al., 2016). With these structural changes in place, we now replace the predictive-error-driven learning rule from above with a direct implementation of Burst-Dependent Plasticity (BDP, Eq 18) (Payeur et al., 2021).

Feedforward excitatory synaptic weights onto pyramidal neurons were updated using the recently proposed burst-propagation rule, implemented at every time step except for validation trials: (Payeur et al., 2021)

$$
\begin{aligned}
\frac{dw_{ij}}{dt} &= \eta\{[\underbrace{B_i}_{\text{Burst?}} -(\underbrace{\bar{P}_i}_{\text{P(Burst)}})\underbrace{E_i}_{\text{event?}}] \quad \underbrace{\tilde{E}_j}_{\text{pre eligibility trace}} \quad \} \\
\bar{E}_i(t) &= \frac{1}{\tau_{avg}} \int_0^\infty E_i(t-\tau) e^{-\tau/\tau_{avg}} d\tau \\
\bar{B}_i(t) &= \frac{1}{\tau_{avg}} \int_0^\infty B_i(t-\tau) e^{-\tau/\tau_{avg}} d\tau \\
\bar{P}_i(t) &= \frac{\bar{B}_i(t)}{\bar{E}_i(t)}
\end{aligned}
\tag{18}
$$

Where E_i is 1 when the neuron emits either a single spike, or the second spike within a 16ms window, and B_i is 1 only for the second spike during a 16ms window. Thus $\bar{E}_i(t)$ and $\bar{B}_i(t)$ are time-weighted averages of the 'event' and 'burst' rates. $\bar{P}_i(t)$ is therefore the probability that an event is the second spike of a burst.

Interpretation Intuitively, this learning rule is a modified pre-post product rule, where postsynaptic term (typically a Boolean indicating a spike) is replaced by $B_i - \hat{P}_i E_i$. This term means that weights will increase if the postsynaptic neuron fires the second spike of a burst, and decrease when the postsynaptic neuron fires a singlet or first spike of a burst. Because $\hat{P}$ is primarily driven by the distal dendritic inputs (Naud and Sprekeler, 2018), this means that the feedback term is responsible for driving learning. We can directly compare this spike-based model to our gated-and-error-driven associative rule from the continuous model. Both equations contain a short-timescale representation of presynaptic activity, either as the low-passed activation ($R_{pre}(t)$ in the PM), or as a synaptic spike-trace ($E_j(t)$ in BDP). The feedback gating term from PM ($D(t)$) is analogous to the Burst Rate (B) in BDP. The most difficult analogy is how the error driven term from the predictive module $(D(t) * (v(t) - v(t-1)))$ relates to the error-driven aspect of BDP $(B_i - P_i * E_i)$. However, one can see the mapping if we consider the BDP framework with a small population of neurons all of which have nearly-balanced distal dendritic inputs. This BDP error term can then be approximated as $F(D(t), v(t)) - D(t)_{lowpass} * v(t))$, where F incorporates the slight nonlinear effect of distal and basal potentials on burst probability. This revised BDP now resembles the error term from the predictive module, by incorporating the difference between an instantaneous and low-passed activity rate.

Readouts were performed by convolution of the recorded spike-train of interest with a synaptic kernel (Fig 4C). A linear readout for this convolved signal was then created by minimal least-squares linear weighting (Eq 21). We note that this readout term, and corresponding error terms, are never provided to the model, and are only used for interpreting the function of the network.

3 Results & Discussion

3.1 Overall Model Comparisons

We begin with a high level summary of means-square error performance across all models and identification of hyperparameters. Table 4 shows a summary of best hyperparameter sets for each model type, identified by minimum validation (full rollout/autonomous) error at any point in training. Trends for each model-types best parameter set are described in their corresponding sections below. Overall, the LSTM and baseline Elman RNN models show the lowest local error, supporting the general claim that these machine learning models, which utilize backpropogation through time, are highly accurate at predicting dynamics in the short term. However, when evaluating the full rollout condition, the leaky-integrator approach outperforms the LSTM, and this performance is further improved by incorporating the novel learning rule of the predictive module.

Model Base	Hyperparameters	Min Train Error	Min Val Error
Baseline RNN	Units: 32, **64**, 128 Depth: **1**, 2, 3, 4 Learning Rate: 10^{-2}, $\mathbf{10^{-3}}$, 10^{-4}	0.004	0.144
LSTM	Units: 32, **64**, 128 Depth: **1**, 2, 3, 4 Learning Rate: 10^{-2}, $\mathbf{10^{-3}}$, 10^{-4}	**0.002**	0.119
Stacked RNN	Units: 32, **64**, 128 Depth: **2**, 3, 4 Learning Rate: 10^{-2}, $\mathbf{10^{-3}}$, 10^{-4}	0.063	0.125
Leaky Integrator	Units: 32, **64**, 128 Depth: 2, **3**, 4 Learning Rate: 10^{-2}, $\mathbf{10^{-3}}$, 10^{-4}	0.060	0.082
Separate FF and FB	Units: 32, **64**, 128 Depth: 2, **3**, 4 Learning Rate: 10^{-2}, $\mathbf{10^{-3}}$, 10^{-4}	0.108	0.109
Predictive Module	Units: 32, **64**, 128 Depth: 2, **3**, 4 Learning Rate: $\mathbf{10^{-2}}$, 10^{-3}, 10^{-4}	0.008	**0.027**
Spiking Model	N/A	0.067	0.070

Table 4. Summary of performance across all tested models, including hyperparameters searched. Each row shows a model class, as separated out in the methods section. The bolded parameters in the second column indicate the best hyperparameter set, chosen by minimum validation error. The best models for local error (LSTM) and validation error (predictive module) are identified by bolding of their MSE.

3.2 Baseline Model

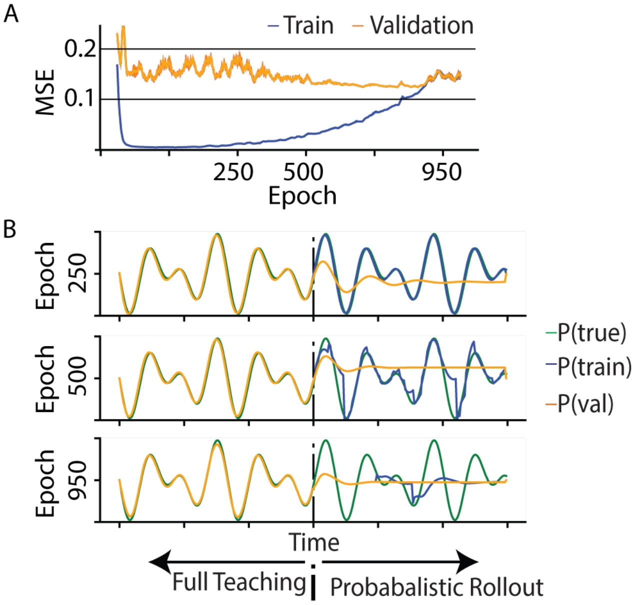

Fig 5. Performance of the baseline RNN **A** Performance over training for both the local training ratio (blue) and validation ratio of zero (orange) as teaching ratio decreases. Initially, local loss approaches zero and the model is able to very accurately predict the next input, given the ground truth. As training progresses, both the partially driven and rollout system approach the same performance. **B** Showing example outputs for the partially driven (blue), and full rollout (orange) systems compared to the target signal (green) at three points in training. (Top) Early in training, the network predicts the next input accurately, given the ground truth, but decays to zero when run in full rollout mode. (Middle) Midway through training, the network can predict a few timesteps into the future accruing noise (blue) and snap back to the correct trajectory when a teaching frame occurs. At this point, the full rollout configuration will tend to maintain an output close to the last observed value. (Bottom) Late in training, the network is severely dampened, and will quickly create a constant-zero output, changing briefly when teaching frames occur (blue shifts away from orange for periods when teaching appears).

This network initially learns to predict the next input with a high degree of accuracy, but fails to predict in an undriven state (Fig 5 A). As training progresses, the network continues to learn short-term predictions (Fig5 B, middle). During this stage of training the BPTT network has learned some of the longer term dynamics, and will poorly fit 1-2 cycles before decaying to a constant output (orange trace). At this point small sequences of frames without the ground truth tend to cause network activity to diverge from the target, but quickly return when the external stimulus is presented again. In the final stages of learning the BPTT network performs very similarly to the full-rollout case from partway through training. From these examples we conclude that while the baseline model is able to accurately act as a predictive autoencoder, the network is never able to predict beyond a handful of frames faithfully.

3.3 Intermediate Models

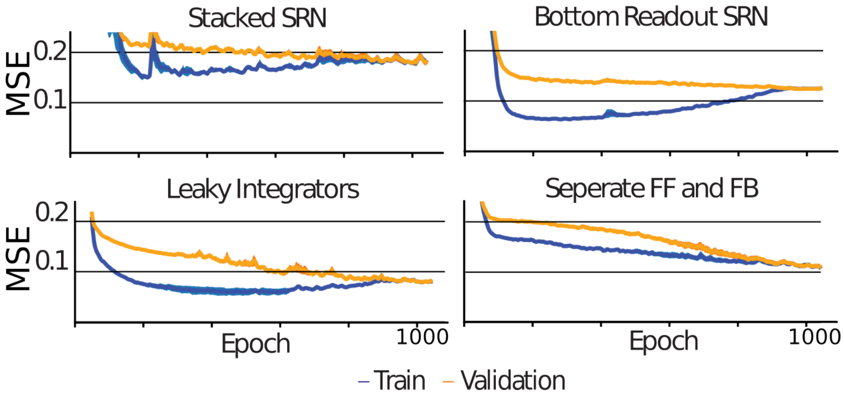

Fig 6. Intermediate Models, showing stepwise changes in overall performance. **Top Left** Beginning with a stacked bidirectionally connected RNN, but reconstructing from the 'top' layer, BPTT is unable to converge even for largely externally driven (early blue) cases. **Top Right** Continuing with a stacked RNN, but now reconstructing from the activity of the lowest layer, the network is more closely able to match the performance of the baseline model. This suggests that having sensory-driven feedforward inputs is critical to utilizing the hidden representations derived by other regions. **Bottom Left** By replacing the RNN dynamics with leaky integrators both the probabilistic and full rollout conditions improve. The training performance (blue) still does not perform as well as baseline models, but the full rollout condition (orange) now more closely matches the external dynamics. **Bottom Right** The introduction of separate feedforward and feedback pathways, while anatomically realistic, decreases performance for both training and validation forcing.

Investigating the learning curves of the intermediate models gives us a high level understanding of what architectural changes the backpropogation approach is able to make use of. In the stacked RNN (Fig 6) performance on both the teacher and full rollout conditions is significantly lower than the performance of the baseline RNN. This suggests that the reciprocal connections interfere with BPTT over long periods, consistent with the observation that the baseline model performed best with a lower number of layers. When utilizing the same bidirectional connectivity and reading out from the lowest layers ('Bottom Readout RNN') however, the short-term prediction increases significantly, though still higher than the baseline. The fully autonomous mode though now has a minimum error that is lower than the baseline, suggesting the the bottom readout approach allows useful information from the hidden layers to integrate into the primarily externally driven signal without disrupting the overall dynamics.

Continuing to utilize a bottom-readout approach and replacing the RNN dynamics with a leaky integrator further decrease the validation-mode error (MSE 0.082), far below the minimum achieved in the baseline models (MSE 0.119). However, the speed with which teacher learning occurs is significantly lower, as indicated by the slower decent of the blue line in Figure 6. When introducing separate pathways for feedforward and feedback information, performance was significantly worse for both the local forcing (MSE 0.108) and full rollout (MSE 0.109) cases, where the system primarily learned only to replicate the first frequency component of the target signal. Similar to the initial findings of the top-readout stacked RNN, this suggests that the BP based method is not able to utilize the additional pathway.

3.4 Predictive Module

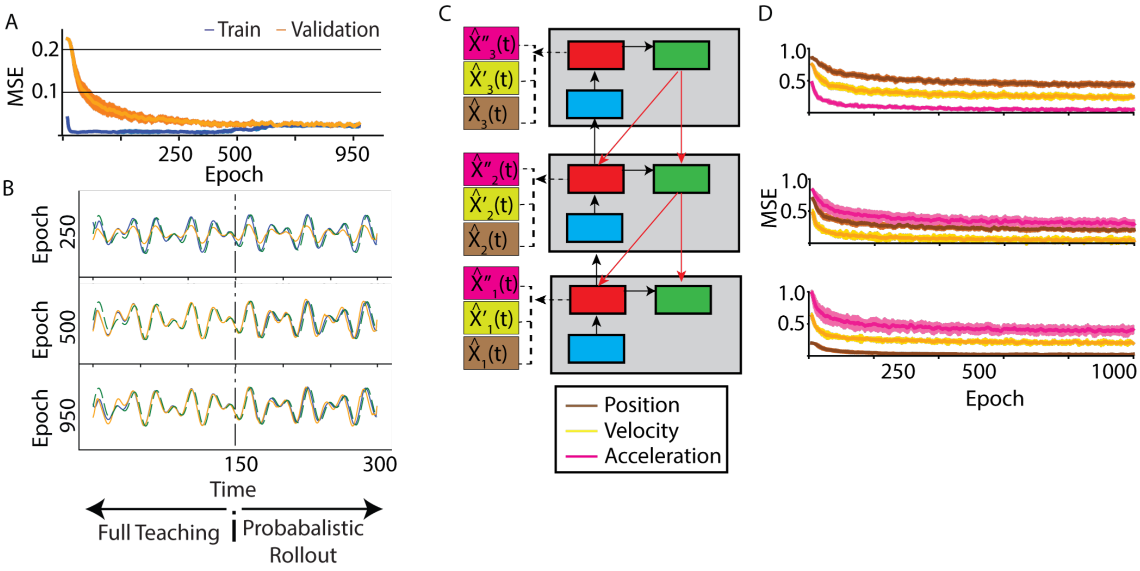

Fig 7. Performance of the predictive module approach. **A** Overall performance over training. The training-forcing condition (blue) quickly approaches performance similar to the backprop-based methods, showing that the lowest region quickly encodes the next-frame location. Compared to the baseline and intermediate models however, the validation (fully autonomous) mode also increases in accuracy quickly, showing that the network is learning to follow the external dynamics, rather than a weighted history of recent activity. **B** Example traces at early, mid, and late stages of training. In the early condition (top) the conditions match closely, though the autonomous mode has a tendency to undershoot local peaks. By mid training (middle) the autonomous mode now closely follows the ground truth. In late training (bottom) the predictions continue to match the target. **C** Decoding additional signals from each region of the model. (Left) Brown, yellow, and pink boxes indicate separate decoders trained for position, acceleration, and velocity for each of the regions, for a total of nine separately trained decoders. **D** Right Shows the accuracy for each of these decoders over the course of training. The lowest region most strongly codes for position (brown), with decreasing accuracy for decoding higher derivatives. The intermediate region can most strongly decode velocity (yellow), while the highest has the strongest tuning for acceleration (pink).

Under the predictive module framework, the network is able to quickly learn to reproduce the next observed frame (Fig 7 A, blue), but initially does not predict well without external drive. However, within the first 200 epochs, the network begins to faithfully predict trajectories in both the partially-driven and undriven states, with a slight undershooting of local peaks in the undriven state. Over the course of training the partially-driven error continues to stay low, despite a decreasing rate of external stimulus, while autonomous error approaches similar performance (down to 0.027 minimum autonomous error, down from 0.082 in the leaky-integrator intermediate model). In addition to investigating how well the predictive module network learned to predict long temporal delays, we were interested in what role each of those regions played in the overall learned representations. We attempted to decode not only the position, but also the temporal derivatives of this variable, utilizing three separate decoders for each module Each decoder was optimized by cross-validated least-squares minimization between the ground-truth signal and the supragranular activity on each trial (Eq 21). We found that each module most strongly represented increasing temporal derivatives of position (i.e. distance, velocity and acceleration). This suggests that the later modules are encoding temporal derivatives of lower-level variables, which in turn provide contextual information to lower regions, further increasing predictive power. The multiple transformations between each module's output and the feedback signal allows for learned coefficients and non-linearities in how that feedback signal affects the current prediction.

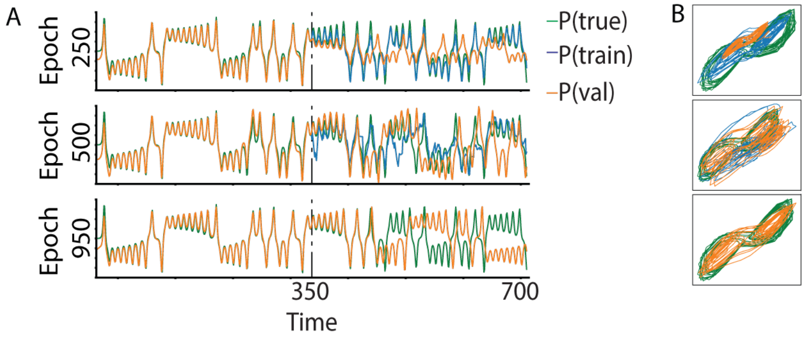

Fig 8. Predictive Module activity on a more complicated task. Performance of the predictive module approach. **A** Showing decoded position along the first dimension at various points in training. At all points the local forcing ratio closely (blue) closely matches the ground truth (orange). Early in training the autonomous dynamics quickly diverge from ground truth, are lower magnitude. In mid training(middle) the autonomous teacher-forcing (blue) will tend to diverge from ground-truth, but comes back when a sample is presented. The autonomous trajectory appears to match the general switching behavior of the Lorenz equations. Later in training (bottom) the autonomous mode stays close to ground truth for long periods before diverging due to chaotic behavior. However, the general behavior still matches the dual-mode oscillations of the ground truth. **B** Showing the first two dimensions at various points in training, only for the second half of the trial. During early training (top) the autonomous trajectory is off center and lower magnitude than ground truth and semi-forcing case. During mid training (middle) the fully autonomous trajectory fills regions that the ground truth does not. Later in training (bottom) the autonomous trajectory more fully matches the 2D regions that the ground truth visits.

In order to test how this model generalizes to more complex tasks we next tested the model on the Lorenz equations, a set of deterministic differential equations known for their sensitivity to initial conditions. As with the sum of sinusoids task, decoding is performed by training on the forced (first half) of a trial, and using the learned weights to decode for the second half of the trial. We find, that by the end of training the network is able to accurately recreate the general attractor dynamics of the task (Fig

8A, bottom). This matches the qualitative results of other methods which aim to fit dynamics rather than explicit input values (Sussillo and Abbott, 2009). Consistent with the nature of the task, the exact trajectories diverge when the internal representation is near the origin and partial derivatives are highly variant between close locations. At the midpoint in training (Fig 8A, middle) the autonomous trajectory similarly matches the general dynamics of the task, but tends to visit locations that the underlying dynamics do not (Fig 8B, middle). However when partial teacher forcing is present, the external stimulus is sufficient to keep the decoded position near the ground truth. For the fully trained model (epoch 950) we attempted to decode temporal derivatives of the *internal* position generated by the position decoding of the lowest region. Consistent with the above results, we found that the velocity strongly represented in the second region but not the first or third. The third region did not strongly encode position, speed, or acceleration terms (MSE greater than 0.14 in all cases).

3.5 Spiking Model

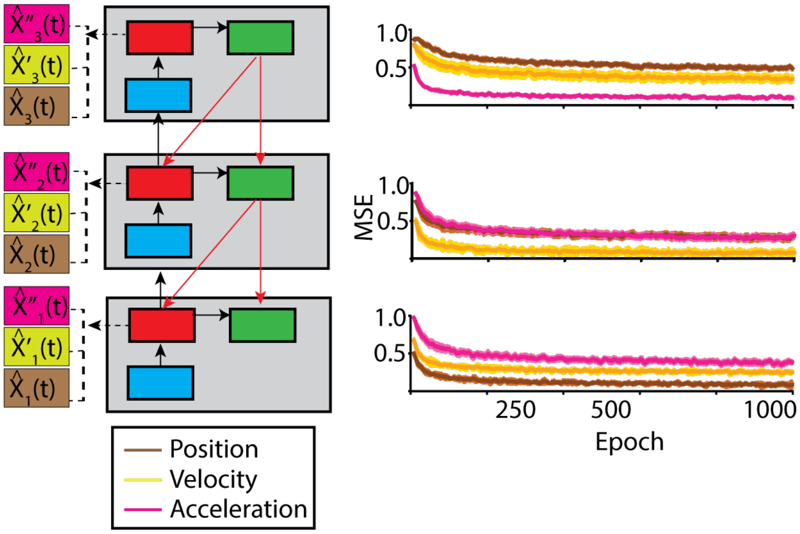

Fig 9. Summary of spiking model decodability. (Left) Showing the readout architecture, identical to the setup for the predictive module. (Right) Showing decodability of position and temporal derivatives over training in superficial pyramidal neurons after convolving by a postsynaptic readout kernel (see methods). The optimized decoders show that the lowest region again encodes position, with velocity being most strongly encoded in the second region, and region 3 most strongly encoding acceleration terms.

As shown in Fig 9, the spiking network showed similar capabilities for function to the predictive module presented in Fig 7. The spiking network is able to accurately predict at long time-scales. The L2/3 neurons in region 1 (bottom) most accurately encodes position, the L2/3 neurons in region 2 (middle) most accurately encode velocity, and the L2/3 neurons in region 3 (top) encode acceleration (Fig 9).

Thus, the region most directly connected to the external input mimics the dynamics of spatiotemporal patterns, while the more distal regions generate abstracted signals which can help guide lower-level activity, but do not directly mimic the inputs.

3.6 Necessity of Biological Components

As discussed above, there are a variety of biologically inspired components of this model which are expected to be necessary for the overall function of the proposed learning rule (Naud and Sprekeler, 2018). Given the success of the full spiking model, we next set to verify that the biological mechanisms are necessary for learning the task, by running two simplified versions and evaluating their performance. Figure 10 shows the results of these experiments.

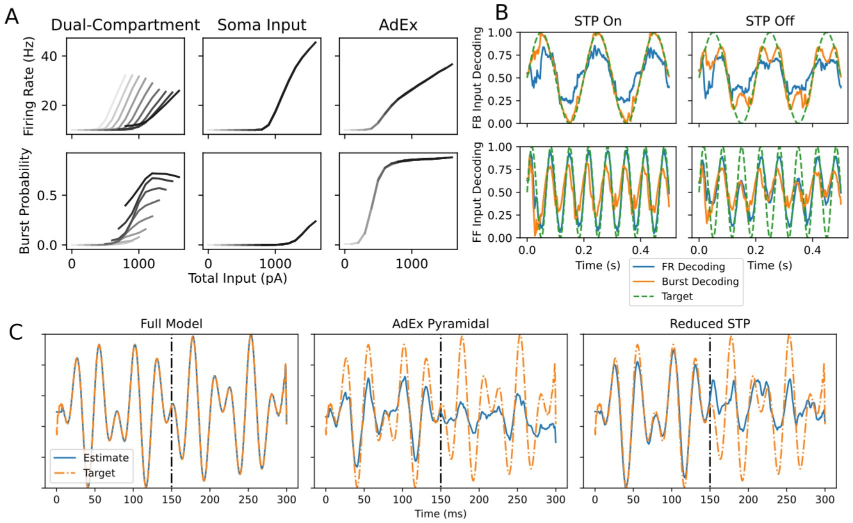

Fig 10. Necessity of biological aspects. **A** Showing the event rate (top) and burst probability (bottom) of various pyramidal neuron models in response to current pulse inputs. (Left Column) In the full dual-compartment model, burst rate is primarily driven by the amount of current form the feedback pathway (darkness of lines, ranging from 0 picoamps (lightest) to 800 picoamps (darkest)). (Middle Column) When the feedforward and feedback pulse both converge on the somatic compartment the event and burst responses are driven only by total input levels (indicated by overlapping lines). (Right Column) In a single compartment adaptive exponential pyramidal neuron the responses are again tuned only to the total input current. **B** Showing the efficacy of short-term plasticity rules. For a single cortical column with the full STP rules (Left) we were able to decode the feedforward input from superficial firing rates (Bottom) and the feedback input from burst rates (Top). When STP rules are removed (Right) from inter-regional connections (table 3) the decodability from the left column is lost. **C** Showing the final behavior of (Left) Fully intact model (Middle) Single compartment modification (Right) Reduced STP dynamics models. Both manipulations fail to follow the external dynamics of the stimulus, confirming the necessity of both mechanisms to drive the burst-dependent learning.

Dual Compartment Neurons The dual compartment model of pyramidal neurons is based on previous implementations (Naud and Sprekeler, 2018), where they show that inputs onto the basal and distal compartments allows independent control of the firing rate and burst probability of these units. As the burst-dependent learning rule relies on the difference in short-term and longer-term burst rate (eq: 18) we expect that this independent control of compartments is necessary for driving successful learning. Figure 10A shows the response of the dual compartment neuron in response to current injections on the basal and distal compartments, and demonstrates that varying these two values results in independently varying firing rate and burst rates. The remaining two columns show the response of the same model, but receiving all input on the basal compartment (middle column) or a single compartment adaptive exponential model (right column) utilizing the same current sweeps. In contrast to the separate basal and distal inputs, both the fully somatic input and single compartment neuron respond only to the total input current, indicating that firing rate and burst rate can no longer be controlled by separate input streams. This suggests that the BDP learning rule will not be able to utilize separate feedforward and feedback signals and will therefore not converge (see "Learning Capabilities" below).

Short-Term Plasticity In the second modification we verify that short-term plasticity mechanisms are necessary for learning in the hierarchical circuit. The STP parameter sets are chosen such that synapses onto feedforward pathways are depressive, thereby filtering for single events rather than bursts, and feedback synapses are facilitating, therefore filtering for burst rather than single spikes (Markram et al., 1998). We verify this by utilizing a single cortical column module and injecting a slowly varying current into the feedforward pathway and a quickly varying input into the feedback pathway, and attempting to decode these variables from the firing rate and burst rate of the superficial pyramidal neurons, utilizing the same synaptic read out and cross-validation approaches as in Figure 4. Figure 10B shows the results of this experiment for the full model and a reduced model in which the short-term plasticity mechanisms have been removed from the relevant synapses (see Table 3). In the full model the two input streams are able to be decoded from the appropriate firing and burst rates, while this is not the case in the reduced model, confirming the role of STP for filtering the two input streams. Because each module is unable to signal separate FF and FB signals without the STP dynamics, we see that the BDP fails when trained in the absence of these mechanisms (see next section).

Learning Capabilities The previous two manipulations show that the biological components are necessary for separately controlling (Dual Compartment Neurons) or transmitting (Short-Term plasticity) feedforward and feedback signals. This suggests that the burst-dependent plasticity trained network will not correctly update synaptic weights in the absence of either of these mechanisms, which we explicitly test in Figure 10C. Here the full model (left column), single compartment modification (middle column) and reduced STP model (right column) are trained on the same task and under the same teacher scheduling conditions as presented above, and the final performance of each model is plotted. The poor performance of both modified networks confirms that dual compartment units as well as appropriate STP mechanisms are necessary for learning to perform the task.

4 Conclusions

We examined how an architecture inspired by connective motifs from the neocortex might predict the spatiotemporal dynamics of external stimuli. Using a continuous firing rate approximation, we found that incorporating a leak-term with a hierarchical network in which reconstruction occurs at the input layers (Fig 6D) significantly improves long-term prediction compared to standard backpropagation approaches. By then incorporating a laminar structure, in which deeper regions gate learning in lower regions, we were able to further increase these predictions (Fig 7). Investigating the activity patterns in various layers of this network, we found that successive regions were predicting temporal derivatives of activity in their inputs, such that deeper layers represent further temporal derivatives of the external stimulus. This finding highlights general patterns of cortical hierarchy which suggest that low-level sensory cortices give rise to higher level cortex which provides less physically grounded representations.

Utilizing a spiking model based on previous work (Naud and Sprekeler, 2018; Payeur et al., 2021), we showed that a biologically plausible learning rule can likewise result in a network that develops intrinsic connectivity that enables prediction of external stimuli, without an extrinsic teaching signal. Instead, the dynamics within each cortical region appear to follow the temporal derivative of their inputs, and must align the dynamics with more abstracted activity from higher cortical regions.

Previous work has investigated how reservoir networks may learn to predict complex time series (Denève et al., 2017) with external negative feedback systems, or by optimized networks (Sussillo and Abbott, 2009) akin to the sequence-to-sequence approach (Boerlin et al., 2013). However, the present work focused on how a potential interaction of multiple cortical regions forming a hierarchy might develop a representation of network dynamics. The arrangement of a hierarchical system not only resulted in an improved performance, but also resulted in a structured organization of temporal information along the hierarchy. Essential to this performance was feedback along the hierarchy, which allowed higher order information to be integrated in superficial neurons of lower cortical regions (Haeusler and Maass, 2007). The different roles of somatostatin and parvalbumin allowed activity to continue stably when external input is absent, as proposed in recent work (Hertäg and Sprekeler, 2020).

Future Directions There were several purposeful simplifications in this study, that can be expanded upon in future studies. Because the focus here was on translating biologically inspired learning rules to a self-supervised temporal prediction setting, we utilized a low-dimensional and deterministic stimulus. Without additional architectural changes, such as convolution or topologically organized connections which are essential for receptive field generation (Gilbert and Li, 2013), the degree to which this network can be scaled up to behaviorally relevant tasks such as object recognition are limited. We also implemented long-term plasticity only on a minority of connections. This is done in order to stay within the realm of self-supervised signals, but neglects the potentiation of inhibitory synapses (Vogels et al., 2011) and granular-terminating connections which may be important for the formation of purely associative subnetworks. Despite these simplifications, the current model shows how a neural system can utilize a hierarchical organization to closely track the dynamics of external stimuli. Future studies may investigate uses of this model in more complex scenarios, which may lead to predictions about the functional phenotypes of cells that arise throughout the cortex.

5 Acknowledgments

This work is supported by the Office of Naval Research MURI N00014-16-1-2832(M.H.) and MURI N00014-19-1-2571 (M.H.) and DURIP N00014-17-1-2304 (M.H.). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

References

Alexander, A. S., Robinson, J. C., Dannenberg, H., Kinsky, N. R., Levy, S. J., Mau, W., Chapman, G. W., Sullivan, D. W., and Hasselmo, M. E. (2020). Neurophysiological coding of space and time in the hippocampus, entorhinal cortex, and retrosplenial cortex. *Brain and Neuroscience Advances*, 4:239821282097287.

Badre, D. and D'Esposito, M. (2009). Is the rostro-caudal axis of the frontal lobe hierarchical? *Nature reviews. Neuroscience*, 10(9):659–669.

Badre, D. and Frank, M. J. (2012). Mechanisms of hierarchical reinforcement learning in cortico-striatal circuits 2: Evidence from fMRI. *Cerebral Cortex*, 22(3):527–536.

Bastos, A. M., Usrey, W. M., Adams, R. A., Mangun, G. R., Fries, P., and Friston, K. J. (2012). Canonical Microcircuits for Predictive Coding. *Neuron*, 76(4):695–711.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Bhandari, A. and Badre, D. (2018). Learning and transfer of working memory gating policies. *Cognition*, 172:89–100.

Bicanski, A. and Burgess, N. (2018). A Neural Level Model of Spatial Memory and Imagery. *eLife*, 7, e33752. tex.ids= bicanski_burgess_2018a.

Bittner, K. C., Milstein, A. D., Grienberger, C., Romani, S., and Magee, J. C. (2017). Behavioral time scale synaptic plasticity underlies CA1 place fields. *Science*, 357(6355):1033–1036.

Boerlin, M., Machens, C. K., and Denève, S. (2013). Predictive Coding of Dynamical Variables in Balanced Spiking Networks. *PLoS Computational Biology*, 9(11):e1003258.

Brandon, M. P., Bogaard, A. R., Schultheiss, N. W., and Hasselmo, M. E. (2013). Segregation of cortical head direction cell assemblies on alternating theta cycles. *Nature Neuroscience*, 16(6):739–748.

Brea, J., Gaál, A. T., Urbanczik, R., and Senn, W. (2016). Prospective Coding by Spiking Neurons. *PLoS Comput Biol*, 12(6):1005003.

Buschman, T. J., Denovellis, E. L., Diogo, C., Bullock, D., and Miller, E. K. (2012). Synchronous oscillatory neural ensembles for rules in the prefrontal cortex. *Neuron*, 76(4):838–46.

Byrne, P., Becker, S., and Burgess, N. (2007). Remembering the Past and Imagining the Future: A Neural Model of Spatial Memory and Imagery. *Psychological Review*, page 36.

Dayan, P., Hinton, G. E., Neal, R. M., and Zemel, R. S. (1995). The Helmholtz Machine. *Neural Computation*, 7(5):889–904.

Denève, S., Alemi, A., and Bourdoukan, R. (2017). The Brain as an Efficient and Robust Adaptive Learner. *Neuron*, 94(5):969–977.

Desimone, R. and Schein, S. J. (1987). Visual properties of neurons in area V4 of the macaque: sensitivity to stimulus form. *Journal of Neurophysiology*, 57(3):835–868.

DiCarlo, J., Zoccolan, D., and Rust, N. (2012). How Does the Brain Solve Visual Object Recognition? *Neuron*, 73(3):415–434.

Douglas, R. J., Martin, K. A., and Whitteridge, D. (1989). A Canonical Microcircuit for Neocortex. *Neural Computation*, 1(4):480–488.

Duggins, P. and Eliasmith, C. (2022). Constructing functional models from biophysically-detailed neurons. *PLOS Computational Biology*, 18(9):e1010461.

Gilbert, C. D. and Li, W. (2013). Top-down influences on visual processing. *Nature Reviews Neuroscience*, 14(5):350–363.

Greedy, W., Zhu, H. W., Pemberton, J., Mellor, J., and Costa, R. P. (2022). Single-phase deep learning in cortico-cortical networks. *Advances in Neural Information Processing Systems*. Publisher: arXiv Version Number: 1.

Gregor, K., Danihelka, I., Mnih, A., Blundell, C., and Wierstra, D. (2014). Deep AutoRegressive Networks. *arXiv:1310.8499 [cs, stat]*. arXiv: 1310.8499.

Haeusler, S. and Maass, W. (2007). A statistical analysis of information-processing properties of lamina-specific cortical microcircuit models. *Cerebral Cortex*, 17(1):149–162.

Hasselmo, M. E. (2005). What is the function of hippocampal theta rhythm?—Linking behavioral data to phasic properties of field potential and unit recording data. *Hippocampus*, 15(7):936–949.

Hasselmo, M. E., Rolls, E. T., and Baylis, G. C. (1989). The role of expression and identity in the face-selective responses of neurons in the temporal visual cortex of the monkey. *Behavioural Brain Research*, 32(3):203–218.

Hasselmo, M. E. and Stern, C. E. (2018). A network model of behavioural performance in a rule learning task. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1744).

Hazan, H., Saunders, D. J., Khan, H., Patel, D., Sanghavi, D. T., Siegelmann, H. T., and Kozma, R. (2018). BindsNET: A Machine Learning-Oriented Spiking Neural Networks Library in Python. *Frontiers in Neuroinformatics*, 12.

Hertäg, L. and Sprekeler, H. (2020). Learning prediction error neurons in a canonical interneuron circuit. *eLife*, 9:e57541.

Hinman, J. R., Brandon, M. P., Climer, J. R., Chapman, G. W., and Hasselmo, M. E. (2016). Multiple Running Speed Signals in Medial Entorhinal Cortex. *Neuron*, 91(3):666–679.

Kingma, D. P. and Dhariwal, P. (2018). Glow: Generative Flow with Invertible 1x1 Convolutions. *arXiv*.

Kingma, D. P. and Welling, M. (2019). An Introduction to Variational Autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392. arXiv: 1906.02691.

Koechlin, E. and Summerfield, C. (2007). An information theoretical approach to prefrontal executive function. *Trends in Cognitive Sciences*, 11(6):229–235.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 69:59–69.

Kropff, E., Carmichael, J. E., Moser, E. I., and Moser, M.-B. (2021). Frequency of theta rhythm is controlled by acceleration, but not speed, in running rats. *Neuron*, 109(6):1029–1039.e8.

Kropff, E., Carmichael, J. E., Moser, M.-B., and Moser, E. I. (2015). Speed cells in the medial entorhinal cortex. *Nature*.

Larkum, M. (2013). A cellular mechanism for cortical associations: An organizing principle for the cerebral cortex. *Trends in Neurosciences*, 36.

Lillicrap, T. P., Cownden, D., Tweed, D. B., and Akerman, C. J. (2016). Random synaptic feedback weights support error backpropagation for deep learning. *Nature Publishing Group*, 7.

Lotter, W., Kreiman, G., and Cox, D. (2017). Deep Predictive Coding Networks for Video Prediction and Unsupervised Learning. *arXiv:1605.08104 [cs, q-bio]*. arXiv: 1605.08104.

Lotter, W., Kreiman, G., and Cox, D. (2020). A neural network trained for prediction mimics diverse features of biological neurons and perception. *Nature Machine Intelligence*, 2(4):210–219.

Magee, J. C. and Grienberger, C. (2020). Synaptic Plasticity Forms and Functions. *Annual Review of Neuroscience*, 43(1):annurev–neuro–090919–022842.

Markram, H., Wang, Y., and Tsodyks, M. (1998). Differential signaling via the same axon of neocortical pyramidal neurons. *Proceedings of the National Academy of Sciences*, 95(9):5323–5328.

McNaughton, B. L., Battaglia, F. P., Jensen, O., Moser, E. I., and Moser, M.-B. (2006). Path integration and the neural basis of the 'cognitive map'. *Nature Reviews Neuroscience*, page 16.

Mountcastle, V. (1997). The columnar organization of the neocortex. *Brain*, 120(4):701–722.

Naud, R., Marcille, N., Clopath, C., and Gerstner, W. (2008). Firing patterns in the adaptive exponential integrate-and-fire model. *Biological Cybernetics*, 99(4-5):335–347.

Naud, R. and Sprekeler, H. (2018). Sparse bursts optimize information transmission in a multiplexed neural code. *Proceedings of the National Academy of Sciences*, 115(27):E6329–E6338. tex.ids: naud_sprekeler_2018a.

Nicola, W. and Clopath, C. (2017). Supervised learning in spiking neural networks with FORCE training. *Nature Communications*, 8(1).

O'Keefe, J. and Burgess, N. (2005). Dual phase and rate coding in hippocampal place cells: Theoretical significance and relationship to entorhinal grid cells. *Hippocampus*, 15(7):853–866.

O'Keefe, J. and Dostrovsky, J. (1971). The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. *Brain Research*, 34(1):171–175.

O'Reilly, R. C. (1997). The LEABRA model of neural interactions and learning in the neocortex. *Dissertation Abstracts International: Section B: The Sciences and Engineering*, 57:6792.

O'Reilly, R. C., Russin, J. L., Zolfaghar, M., and Rohrlich, J. (2021). Deep Predictive Learning in Neocortex and Pulvinar. *Journal of Cognitive Neuroscience*, 33(6):1158–1196.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. arXiv:1912.01703 [cs, stat].

Payeur, A., Guerguiev, J., Zenke, F., Richards, B. A., and Naud, R. (2021). Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *Nature Neuroscience*.

Rao, R. P. N. and Ballard, D. H. (1999). Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87.

Recanatesi, S., Farrell, M., Lajoie, G., Deneve, S., Rigotti, M., and Shea-Brown, E. (2021). Predictive learning as a network mechanism for extracting low-dimensional latent space representations. *Nature Communications*, 12(1):1417.

Rockland (2010). Five points on columns. *Frontiers in Neuroanatomy.*

Rumelhart, D. E. and McLelland, D. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition.* MIT Press.

Sussillo, D. and Abbott, L. (2009). Generating Coherent Patterns of Activity from Chaotic Neural Networks. *Neuron*, 63(4):544–557. tex.ids= sussillo_abbott_2009a.

Sutskever, I., Hinton, G., and Taylor, G. (2009). The Recurrent Temporal Restricted Boltzmann Machine. *Advances in Neural Information Processing Systems*, page 8.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *Neural Information Processing*, page 9.

Sutton, R. S., Precup, D., and Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211.

Vercruysse, F., Naud, R., and Sprekeler, H. (2021). Self-organization of a doubly asynchronous irregular network state for spikes and bursts. preprint, Neuroscience.

Vogels, T. P., Sprekeler, H., Zenke, F., Clopath, C., and Gerstner, W. (2011). Inhibitory Plasticity Balances Excitation and Inhibition in Sensory Pathways and Memory Networks. *Science*, 334(6062):1569–1573.

Wallis, J. D., Anderson, K. C., and Miller, E. K. (2001). Single neurons in prefrontal cortex encode abstract rules. *Nature*, 411(6840):953–956.

Williams, R. J. and Zipser, D. (1989). A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1(2):270–280.

Yoo, S. B. M., Tu, J. C., Piantadosi, S. T., and Hayden, B. Y. (2020). The neural basis of predictive pursuit. *Nature Neuroscience*.

Zhu, H., Paschalidis, I. C., and Hasselmo, M. E. (2018). Neural circuits for learning context-dependent associations of stimuli. *Neural Networks*.

6 Supplemental Materials A: Detailed Methods

6.1 Task Parameters

Param	Sin1 Lower	Sin1 Upper		Sin2 Lower	Sin2 Upper
a	1	1		0.5	2.0
f	0.15	0.30		1.5 * f1	2.0 * f1
p	-pi	pi		-pi + p1	pi + p1

Table 5. Summary of parametric generation of task

6.2 Predictive Module

6.3 Spiking Model

6.3.1 AdEx Dynamics

$$\begin{aligned} C\frac{dV}{dt} &= -g_L(V - E_L) - g_L\Delta_T \exp(\frac{V - V_T}{\Delta_T}) + I - w \\ \tau_w \frac{dw}{dt} &= a(V - E_L) - w \end{aligned} \tag{19}$$

$$V \geq V_{Th} \rightarrow \begin{cases} V(t_+) & = V_{reset} \\ w(t_+) & = w(t_-) + b \end{cases} \tag{20}$$

Parameter	Somatostatin Neurons	Parvalbumin Neurons	L4 Stellate
C (pF)	100	100	200
g_L (nS)	5	5	12
E_l (mV)	-70	-70	-70
V_T (mV)	-62	-48	-50
Δ_T (mV)	4	2	2
a (nS)	-.5	0	2
τ_w (ms)	100	1	300
b	10	0	60
V_r (mv)	-65	-55	-58
τ_s (ms)	10	10	5

Table 6. Stellate and interneuron parameters

Parameter	Soma	Dendrite	Description
C	370pF	170pF	Membrane Capacitance
τ	7ms	16ms	Membrane time constant
E_L	-70mV	-70mV	Reversal potential
g	1200pA	1300pA	Passive coupling from gated dendritic channels
τ_w	30ms	100ms	Recovery variable time constant
c	N/A	2600pA	Strength of backpropogating action potential
a_w	N/A	-13pA	Strength of passive leak current in dendrites

Table 7. Pyramidal Neuron parameters. Soma column corresponds to Eq 9, and dendrite column to values in Eq 12

Type	D (s)	F (s)	U	f
EE	1.1	.05	.5	.1
EI	.125	1.2	.05	.1
II	.144	.06	.32	.1
IE	.7	.02	.25	.1
F	.1	.1	.02	.1
D	.02	1	0.9	.1

Table 8. Parameter sets used for the short-term dependent plasticity.

Compartment	μ(pA)	τ_{ou}(ms)	σ(pA)
Somatostatin	0	2	450
Parvalbumin	0	2	450
Stellate	0	2	450
Pyramidal Soma	70	2	450
Pyramidal Dend	-270	2	450
Region 3 Dend	-23	2	450

Table 9. Parameters for the OU noisy input. The final region distal dendrites received different input in order to compensate for the lack of feedback activity and maintain a burst rate similar to other regions.

6.3.2 Synaptic Readout

Spiking models were decoded by convolving a given spike train with an idealized dual-exponential post-synaptic kernel:

$$I_{readout}(t) = \frac{(t-\delta)}{\tau_r} * e^{-(t-\delta)/\tau_r} \tag{21}$$

Where τ_r is the synaptic readout time constant of 10ms. On each trial, the readout weights (W) were optimized by constrained least-squares (‘Ridge’ in sklearn) such that the error term:

$$E = \sum (Y(t) - WI_{readout}(t))^2 + \lambda ||W||_2^2 \tag{22}$$

was minimal, where $Y(t)$ is the signal we are attempting to decode (eg: P(t), P’(t),...), and λ is a constant (0.01) which penalizes large readout weights. These same readout weights are then used during the probabilistic rollout to create the decoded rollout signal. Readout weights are optimized for each epoch.

7 Supplemental Materials B: FORCE

As a second type of baseline, we implemented FORCE training. This type of training, and its resulting outputs, differ greatly from BPTT approaches, and provide additional insight as to how a network may fail at the presented tasks. This network consists of a single recurrent network of leaky integrators, and a linear readout. At each point in time, the value from the readout node is fed back into the network according to a random weight matrix, similar to the rollout mode in the BPTT models. Notably, the only weights that adjust are the weights from the readout node back into the network, thus reducing the number of learned parameters compared to the BPTT approach. These readout weights are adjusted quickly according to the recursive-least-squares (RLS) learning rule. During the rollout period this network performs similarly for both local and teacher ratios. This is because the tight control between the readout term and network activity is directly optimizing the 'autonomous' mode of function. However, as learning progresses and and the network must operate in the presence of unreliable teaching signals, both the local and validation errors increase significantly. This change occurs because the network is able to drift slightly during the autonomous periods, resulting in an inaccurate correlation of presynaptic activity (Eq 24) which results in RLS driving learned readout weights incorrectly. This illustrates how a network that may perform well initially, even at full rollout, degrades in the presence of unreliable external signals.

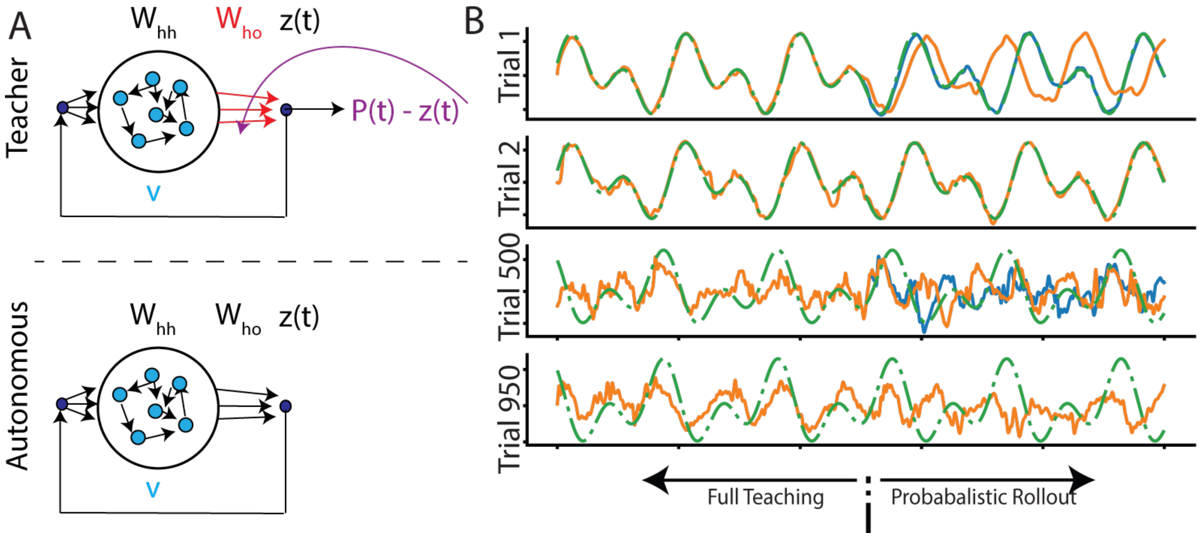

Fig 11. FORCE Training. **A** Setup for training of FORCE. During teacher frames (top) the readout value z(t) is subtracted from the goal readout P(t). This error signal is used to adjust the readout weights W_{ho} During autonomous frames, the system evolves the same as during teaching, but the readout weights are not updated. **B** Example of training FORCE networks. (Top) During the first trial, the network is able to closely follow the inputs signal, due to the quick adjustment of readout weights. During full autonomous (orange) however, the network begins to diverge, indicating that the 150 frames presented during full teacher are not sufficient to train the readout weights. During the second trial (Second row) the network is able to perform both the probabilistic autonomous and full autonomous conditions. However, as training progresses (third and bottom row) the network produces a noisy and chaotic output. The unreliable teaching signal has driven the readout weights and correlation matrix from their initially tuned regime, and even full teaching frames can not bring the network back to reliable readout.

Units in the FORCE network are simple recurrent networks with a leak term, and a linear readout. The update of the system was defined as:

$$\begin{aligned}\tau\frac{dv}{dt} &= -v(t) + W_{hh}r(t) + W_{ih}I(t) + W_{zh}Z(t-1)\\ r &= \tanh(v(t))\\ z(t) &= W_{ho}r(t)\end{aligned} \tag{23}$$

Where r(t) represents the “firing rate” of the hidden units, z(t) the activity of the readout units, and I(t) represents the task-dependent input signal. At every time step a running average of presynaptic correlations was updated:

$$Q(t) = Q(t-1) - \frac{Q(t-1)r(t)r^T(t)Q(t-1)}{1 + r^T Q(t-1)r(t)} \tag{24}$$

On time steps where learning occurred,readout weights were updated according to the recurrent least squares (RLS) algorithm:

$$\begin{aligned}e(t) &= z(t) - y(t)\\ W_{ho}(t) &= W_{ho}(t-1) - e(t)Q(t)r(t)\end{aligned} \tag{25}$$

Weights were initialized and parameters set as standard (Sussillo and Abbott, 2009) and reported in table 10, Weights were initialized according to a Gaussian distribution, and clamped to zero with a probability 1-p (see table for values).

τ	10ms
N_I	2
N_h	1000
N_z	2
W_{ih}	$Gauss(0, \frac{1}{\sqrt{N_I}})$, p=0.1
W_{hh}	$1.5 * Gauss(0, \frac{1}{\sqrt{N_h}})$, p=0.1
W_{hz}	0, p=1
W_{zh}	$Unif(-1, 1)$, p=1

Table 10. Parameters used for FORCE network. Weight parameters indicate the probability distribution that initial values were drawn from.